

In the Claims

Please amend the claims as follows:

1 1. (Currently Amended) A processor having a changeable  
B6 architected state, comprising:

3 instruction memory for storing instructions;  
4 an instruction pipeline, wherein an instruction which passes  
5 entirely through the pipeline alters the architected state and  
6 wherein the pipeline comprises circuitry for fetching instructions  
7 from the instruction memory into the pipeline;

8 circuitry for storing an annul code having an annul bit  
9 corresponding to each instruction of a group of instructions in the  
10 pipeline; and

11 circuitry for preventing one or more selected instructions in  
12 the group from altering the architected state in response to the  
13 corresponding annul bit of the annul code.

1 2. (Original) The processor of claim 1:  
2 wherein the instruction pipeline further comprises a plurality  
3 of execution units; and  
4 wherein one of more the plurality of execution units receives  
5 a corresponding instruction for executing the corresponding  
6 instruction in a given clock cycle.

1 3. (Currently Amended) The processor of claim 2:  
2 wherein each annul bit of the annul code comprises ~~a plurality~~  
3 ~~of bit states~~ either of a first state or a second state;  
4 wherein the circuitry for preventing one or more selected  
5 instructions in the group from altering the architected state  
6 comprises circuitry for coupling ~~the plurality of bit states~~ annul  
7 bits to respective ones of the plurality of execution units;

8 wherein in response to a ~~bit state~~ an annul bit being a first  
9 state the execution unit to which the annul bit ~~state~~ is coupled  
10 does not execute the corresponding instruction in the given clock  
11 cycle; and

12 wherein in response to a ~~bit state~~ an annul bit being a second  
13 state different than the first state the execution unit to which  
14 the annul bit ~~state~~ is coupled does execute the corresponding  
15 instruction in the given clock cycle.

1 4. (Original) The processor of claim 3 wherein the plurality  
2 of execution units comprises a load/store unit, a multiply unit, an  
3 ALU unit, and a shift unit.

1 5. (Currently Amended) The processor of claim 3:  
2 wherein the plurality of execution units are operable such  
3 that in a given clock cycle an integer number N of the plurality of  
4 execution units are scheduled to execute; and  
5 wherein the circuitry for coupling the ~~plurality of bit states~~  
6 annul bits to respective ones of the plurality of execution units  
7 comprises circuitry for coupling only the integer number N of the  
8 ~~plurality of bit states~~ annul bits to the plurality of execution  
9 units which are scheduled to execute in the given clock cycle.

1 6. (Currently Amended) The processor of claim 5:  
2 wherein the instructions corresponding to the annul code  
3 comprise:  
4 a first group of one or more instructions logically  
5 arranged after a conditional instruction and to be executed if the  
6 condition is satisfied;  
7 a second group of one or more instructions logically  
8 arranged after the conditional instruction and to be executed if  
9 the condition is not satisfied;

10 wherein the circuitry for preventing prevents the first group  
11 of instructions from altering the architected state in response to  
12 corresponding annul bits of the annul code having the first state  
13 if the condition is not satisfied; and

Bl 14 wherein the circuitry for preventing prevents the second group  
15 of instructions from altering the architected state in response to  
16 corresponding annul bits of the annul code having the first state  
17 if the condition is satisfied.

1 7. (Currently Amended) The processor of claim 5:  
2 wherein the instructions corresponding to the annul code  
3 comprise instructions corresponding to a software loop scheduled to  
4 execute for an integer M number of iterations; and  
5 wherein during a given iteration the circuitry for preventing  
6 prevents one or more of the instructions corresponding to the annul  
7 bits of the annul code from altering the architected state in  
8 response to the annul bits of the annul code and based on a  
9 relationship of the given iteration to the integer M number of  
10 iterations.

1 8. (Currently Amended) The processor of claim 3:  
2 wherein the instructions corresponding to the annul code  
3 comprise instructions corresponding to software loop scheduled to  
4 execute for an integer M number of iterations; and  
5 wherein during a given iteration the circuitry for preventing  
6 prevents one or more of the instructions corresponding to the annul  
7 bits of the annul code from altering the architected state in  
8 response to the annul bits of the annul code and based on a  
9 relationship of the given iteration to the integer M number of  
10 iterations.

1 9. (Currently Amended) The processor of claim 3:  
2 wherein the group of instructions corresponding to the annul  
3 code comprise:

4 a first group of one or more instructions logically  
5 arranged after a conditional instruction and to be executed if the  
6 condition is satisfied;

7 a second group of one or more instructions logically  
8 arranged after the conditional instruction and to be executed if  
9 the condition is not satisfied;

10 wherein the ~~bit-states~~ annul bits corresponding to the first  
11 group of one or more instructions are set to the first state and  
12 the ~~bit-states~~ annul bits corresponding to the second group of one  
13 or more instructions are set to the second state if the condition  
14 is not satisfied; and

15 wherein the ~~bit-states~~ annul bits corresponding to the first  
16 group of one or more instructions are set to the second state and  
17 the ~~bit-states~~ annul bits corresponding to the second group of one  
18 or more instructions are set to the first state if the condition is  
19 satisfied.

1 10. (Original) The processor of claim 1 wherein the annul  
2 code is generated in response to one or more constant generating  
3 instructions.

1 11. (Original) The processor of claim 1 wherein the annul  
2 code is loaded from a memory.

1 12. (Original) The processor of claim 1 wherein the annul  
2 code is an immediate value in an instruction passing through the  
3 pipeline.

1 13. (Original) The processor of claim 1 wherein the annul  
2 code comprises 32 bits.

1 14. (Original) The processor of claim 1 wherein the annul  
2 code comprises more than 32 bits.

BL 1 15. (Original) The processor of claim 1:  
2 wherein the annul code comprises 64 bits; and  
3 wherein the annul code is formed in response to two thirty-two  
4 bit values.

1 16. (Currently Amended) The processor of claim 1:  
2 wherein the annul code is loaded in response to an instruction  
3 having a condition predicate;  
4 wherein the annul code comprises a first annul code loaded in  
5 response to the condition predicate being satisfied; and  
6 wherein the annul code comprises a second annul code loaded in  
7 response to the condition predicate not being satisfied.

1 17. (Currently Amended) The processor of claim 16:  
2 and further comprising a first data register and a second data  
3 register;  
4 wherein the first annul code is stored in the first data  
5 register; and  
6 wherein the ~~first~~ second annul code is stored in the second  
7 data register.

1 18. (Currently Amended) The processor of claim 16:  
2 and further comprising a data register;  
3 wherein the first annul code is stored in a first one-half of  
4 the data register; and

5 wherein the second annul code is stored in a second one-half  
6 of the data register different from the first one-half.

1 19. (Original) The processor of claim 1:

2 wherein an annul instruction passing through the pipeline  
3 specifies an integer N; and

4 wherein the annul code is formed in response to the integer  
5 value N such that the circuitry for preventing prevents N  
6 successive instructions in the pipeline from altering the  
7 architected state.

1 20. (Currently Amended) The processor of claim 1:

2 and further comprising a register;

3 wherein the register stores the annul code which comprises a  
4 set of annul bits having a first logical value and a set of annul  
5 bits having a second logical value;

6 wherein the annul code is loaded in response to an instruction  
7 having a condition predicate;

8 wherein the circuitry for preventing prevents instructions  
9 corresponding to ~~the~~ annul bits having a first logical value from  
10 altering the architected state in response to the condition  
11 predicate being satisfied; and

12 wherein the circuitry for preventing prevents instructions  
13 corresponding to ~~the~~ annul bits having a second logical value  
14 opposite the first logical state from altering the architected  
15 state in response to the condition predicate not being satisfied.

1 21. (Original) The processor of claim 1 and further  
2 comprising circuitry for storing a portion of the annul code in  
3 response to receipt of an interrupt.

1 22. (New) A method of data processing comprising the steps  
2 of:  
3 identifying a group of instructions including a tree of a  
4 plurality of conditional branch instructions;  
5 for each conditional branch instruction within the group of  
6 instructions  
7 forming a first annul code having an annul bit  
8 corresponding to each instruction following the conditional  
9 branch instruction, the annul bit having a first logical state  
10 for instructions following the detected conditional branch  
11 instruction executed if a condition of the conditional branch  
12 instruction is satisfied and a second logical state opposite  
13 to the first logical state for instructions following the  
14 detected conditional branch instruction executed if the  
15 condition of the conditional branch instruction is not  
16 satisfied,  
17 forming a second annul code having an annul bit  
18 corresponding to each instruction following the conditional  
19 branch instruction, the annul bit having the second logical  
20 state for instructions following the detected conditional  
21 branch instruction executed if a condition of the conditional  
22 branch instruction is satisfied and the first logical state  
23 for instructions following the detected conditional branch  
24 instruction executed if the condition of the conditional  
25 branch instruction is not satisfied;  
26 upon execution of the group of instructions  
27 detecting each conditional branch instruction,  
28 evaluating the condition of the conditional branch  
29 instruction,  
30 loading the corresponding first annul code if the  
31 condition of the conditional branch instruction is satisfied,

32 loading the corresponding second annul code if the  
33 condition of the conditional branch instruction is not  
34 satisfied,  
35 executing each instruction following the conditional  
36 branch instruction if the corresponding annul bit of the  
37 corresponding annul code has the first state, and  
38 not executing each instruction following the conditional  
39 branch instruction if the corresponding annul bit of the  
40 corresponding annul code has the second state.

---